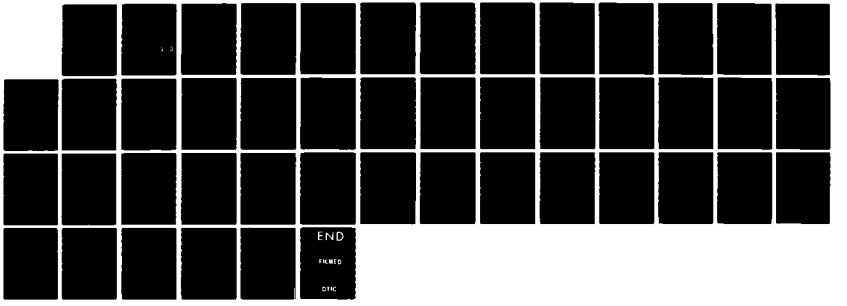
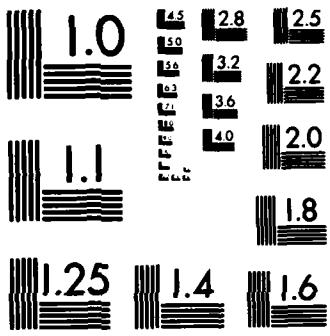
AD-A150 661 SOME RECENT DEVELOPMENTS IN SYSTEMS RELIABILITY(U) 1/1
OKLAHOMA STATE UNIV STILLWATER OFFICE OF BUSINESS AND
ECONOMIC RESEARCH M O LOCKS JAN 85 OSU-OBER-85-1
UNCLASSIFIED AFOSR-TR-85-0094 AFOSR-82-0251 F/G 5/1 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(2)

SOME RECENT DEVELOPMENTS IN SYSTEM RELIABILITY

Comparative study of inclusion-exclusion, sum of disjoint products and topological reliability.

Mitchell O. Locks

OSU OBER 85-1 1/85

Key words: system reliability, inclusion-exclusion, sum of disjoint products, topological reliability, m-out-of-n, source-to-multiple terminal reliability

January 1985 Research Series 85-1

This research has been supported by Air Force Office of Scientific Research under Control Number AFOSR-82-0251 with the College of Business Administration at Oklahoma State University. Distribution is unlimited and is approved for public release.

SOME RECENT DEVELOPMENTS IN SYSTEM RELIABILITY*

0.0 Abstract

System reliability analysis calculates the probability of success for a system, based on the component reliabilities and the configuration. First, a logic function is obtained in the form of either a tree, chart, graph, diagram or list of paths. From this logic function a probability formula is derived. The classical or conventional method of generating a formula is inclusion-exclusion (IE).

With the past decade there have been some significant new developments that resulted in ways to estimate the system reliability that are more efficient than IE. Two of these techniques are discussed in this paper: sum of disjoint products (SDP), and the topological reliability (TR) of Satyanarayana and Prabhaker (S&P). This paper covers the theory and procedures of both techniques, shows their interrelationships with IE, and discusses complexity considerations and computer time needed for preparation of a system formula. The discussion on TR also includes advanced applications such as overall reliability and k-terminal reliability, classes of problems that can conveniently be solved by TR with minor modifications of the logic.

Signals - supplied key words included: → p. 1473

*In preparing this report, I have benefited from discussions with A. Satyanarayana, Stevens Institute of Technology, R. E. Barlow, UC Berkeley, and W. G. Schneeweiss and K. D. Heidtmann, both of Fernuniversitat, Hagen, Germany, D5800.

CONTENTS

1.0 Introduction

System reliability analysis deals with designing and synthesizing networks of elements that function jointly towards a common goal, and estimating the probability of success. Examples of applications of system reliability include: space missions, military missions and hardware development projects, nuclear power generation accident prevention, software reliability, oil drilling, management information systems, communications in complex networks and civilian product R&D and distribution programs.

The system is represented by a graph showing the components and their interconnections and logical interrelationships. A probability formula is derived from the graph; then the component probabilities are substituted into the formula to obtain the probability of success. The best known method of obtaining a formula from a system graph is inclusion-exclusion (IE). It is usually assumed both for simplicity and realism that the system is coherent. The term "coherent" means that success is defined by a set of minimal subsets of elements, all elements in the subset operating successfully; these subsets are called paths, minimal paths or minimal path sets. With IE, the system probability formula is built up recursively, one path at a time.

In the past decade there have been some significant new developments. Fratta and Montanari [4] (1973) published an algorithm for obtaining a system reliability formula by means of sum of disjoint products (SDP). In 1978 Satyanarayana and Prabhaker [10] (S&P) introduced topological reliability (TR) as an alternative to IE to obtain a formula for coherent single-source to single-terminal (s-t) networks, by a search process. In 1979 Abraham [1] published an improved version of SDP. These papers have had a significant impact upon system reliability, because both TR and SDP are more economical ways of generating formulas than recursive IE. As a result, it is now

possible to process larger and more complex systems than heretofore, with less computer time and a smaller memory. The purpose of this paper is to present a description of both SDP and TR, and to show why they are superior to conventional methods.

1.1 Sum of disjoint products

The economies of SDP are due to a simple but important principle: if the terms of the logic function are disjoint with no sets in common, then the logic function is the same as the probability formula. By the addition law of probabilities, the formula is entirely additive.

As with IE, the SDP system formula is obtained from the paths. With IE, however, because sets of joint paths are alternately included and excluded, the terms obtained at every recursive step alternately add and subtract probabilities; since there are as many subtractions as additions, when there is extensive redundancy the polynomial formula can be very long. With SDP, however, the formula is a sum of additive terms only. Experience has shown that an SDP formula for any but very small systems is smaller than IE, and an order of magnitude smaller for a very large system.

1.2 Topological reliability

In their paper on TR, S&P showed that the noncancelling terms of an IE formula are 1:1 with certain subsets of the system graph, called the p-acyclic subgraphs. In TR all p-acyclic subgraphs are identified, without any duplications, by a search process that systematically strips edges so as to find the p-acyclic subgraphs as well as their signs (+ or -) in the system probability formula; at the same time the formula is derived in nested and factored form, equivalent factor by term to the IE polynomial, but with fewer computer operations and less memory needed to derive the formula or to calculate the probability of success.

In addition to the saving in computer time and memory, another advantage of TR is that a wider class of applications is possible than with conventional IE. Undirected networks with two-way edges can be processed as easily as directed networks with only one-way edges by substituting two one-way edges in opposite directions for the two-way edges. Reliability formulas can be obtained for communications networks more complex than s-t, with minor modifications of the techniques and software. For example, certain networks have all terminals communicating with each other; this is called "overall reliability." Another example is having every member of a specified subset communicate with every other member; this is called "k-terminal reliability." In this report examples are provided to show the simple rearrangement that converts an overall problem or a k-terminal system into a graph of acceptable form for TR.

When a new process or technique is first introduced, there can be difficulties in interpreting and using the results, even when it proposes significant improvements over the incumbent way of doing things, because there is a need for a new terminology that has not been fully developed; as a result the explanation can be overly long on some topics and too brief on others. Such is the case with the S&P description of TR. In this report, we have attempted to simplify the description, while preserving the essence of the TR method, making some minor changes in the terminology and omitting the mathematical theorems and proofs.

1.3 Highlights of this report and summary of findings

Our major results are as follows:

1. All three algorithms, conventional IE, SDP and TR, are NP-complete; this means that it cannot be guaranteed in general that a large problem can be solved in polynomial time or linear time, but that it may be possible to do so in certain special cases.

2. Like IE, SDP yields a system reliability formula which is a polynomial in the probabilities of the elements; unlike IE, the terms are disjoint. Experimental results show that for large systems, SDP provides a formula with fewer terms.

3. In the special case of an m-out-of-n system, by ordering the paths in a certain way, the SDP formula is linear in the number of terms and polynomial in the number of variables in the system probability function.

4. We provide proofs of aspects of SDP: the inner and outer loops of the Abraham algorithm, and the counting of computer operations for m-out-of-n system formulas with an ordered set of paths.

5. We describe the special relationship between the system reliability formulas obtained by TR and IE. These two different versions give the same result, but IE obtains a string of polynomial terms, whereas the TR formula is in nested and factored form.

6. This report also includes detailed descriptions and explanations of both the terminology and procedures of topological reliability, as a supplement to the seminal 1978 paper by S&P. The terms include: p-graph, p-acyclic graph, p-cyclic graph, search tree terms and family relationships between nodes of the search tree, neutral sequences, formations and dominations, and the relationship between domination and + or - signs. The processing rules include: the weight restriction, the four search processing rules, and the procedure for backtracking and stopping.

7. Although there are some difficulties in adapting to the new TR terminology and algorithm, it is well worth the effort if large problems are to be processed by a computer. TR has many advantages over IE, including: fewer variable entries in the formula, less computer time, virtually no storage and greater flexibility for processing a wider variety of networks

than the source-to-terminal problems that are discussed in practically all of the system reliability literature that has been published to date. These advantages make up in part for the fact that the algorithm is NP-complete, and make it possible to have large problems run in a reasonable amount of computer time.

2.0 Definition of a system

The reliability of a system differs from that of a component primarily in the way it is assessed. The reliability of a component is evaluated from test data; by contrast, the reliability of a system requires component data and the logical configuration of the elements contributing to either success or failure. System reliability is commonly referred to as "prediction" because the probability of success is estimated before there is sufficient test data to objectively assess what the value of this probability is.

2.1 Network reliability graphs for success

The system is a network graph G_0 of functioning components, elements or variables. In the success orientation, every edge of G_0 is a component and every vertex is a logical connection between two or more components. An edge can also represent successful avoidance of a failure type, a "failure mode" that does not occur.

A string connecting two or more edges is a sequence with all of the components required for success. A parallel connection of two or more edges or sequences between the same two vertices denotes redundancy. An unbroken string from a source vertex s to a terminal vertex t with no failures in the string is system success; this string is called a <u>path</u>, sometimes also minimal path, minimal path set or simple path. The system reliability is the probability that there is at least one path in G_0--the probability that there is at least one path with no component failures.

2.2 Fault trees: network reliability graphs for failure

An alternative to the success oriented graph is a logical diagram of failure dependencies, frequently called a fault tree. The elements of the graph are failure modes. The connections between elements are logic gates, an and-gate denoting joint failure of the elements, or else simultaneous transmission of the effect of failures from lower level gates. Another way of saying this is that an and-gate denotes the effect of failures of redundant elements or subsystems. An or-gate denotes the transmission of the effect of the failure of one or more components or or subsystems, as, for example, a serial connection in a success logic diagram.

With fault trees, instead of just one source and one terminal as in s-t, there are multiple sources, representing all the different component and subsystem failure modes, and one terminal, called the top event, failure of the system. A smallest set of elements for failure is called a cut, sometimes also called minimal cut or minimal cut set; the system reliability is the probability that there are no cuts. It is shown in [6] that the cuts can be derived from the paths, or vice versa. Fault tree reliability analysis has become popular in the past decade, largely as a result of the WASH-1400 report on the safety of nuclear power plants [11], and the concern of the public for nuclear safety.

In this report, the description will be entirely with success oriented graphs rather than with fault trees, because the literatures of SDP and TR both trace paths and success logic, rather than failure logic and cuts.

2.3 Inclusion-exclusion

The conventional way of calculating the reliability of a system is the method of inclusion-exclusion (IE), also known as Poincare's Theorem. IE is derived from first principles in the same way that logicians and probabilis

build Venn diagrams: first add the probabilities of the separate events; then subtract the probability of the joint event. The buildup alternates addition and subtraction of the probabilities of joint events. The result is a probability polynomial with alternating additive and subtractive terms; substitute the component reliabilities into this polynomial and obtain the numerical value of the system probability of success.

2.3.1 The probability equation

The IE system reliability formula is built up recursively using as many steps as there are paths, each path contributing an increment of success probability represented by more terms of the polynomial. For example, if G_0 is serial with just a single path, A, the probability of success P(A) is just a single term, the product of the reliabilities of all the elements of A. Let us suppose there are two alternative paths, A and B. The probability of success, P(A or B), is a polynomial with three terms, the sum of the probabilities of both of the paths, P(A) + P(B), minus the probability of the joint event P(A and B), because the joint event is included in both A and B and would be counted twice if this exclusion were not made. The result is an equation with three terms

$$P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B). \qquad (1)$$

Suppose G_0 has n paths; n is any integer greater than 2. The recursive buildup of the formula is an extension of Equation (1). With n paths, $A_1,\ldots,A_n$, by mathematical induction the reliability is

$$P(A_1 \text{ or} \ldots \text{or } A_n) = P(A_1 \text{ or} \ldots \text{or } A_{n-1}) + P(A_n) - P(A_1 \text{ and} \ldots \text{and } A_n).$$

It is shown in Reference [3] that the system equation has the following form: let S_1 be the sum of the probabilities of the n paths, S_2 the sum of the joint probabilities of the n paths taken two at a time, S_3 the sum of the joint probabilities of the paths three at a time, etc. The system reliability is

$$R = S_1 - S_2 + S_3 - \ldots + (-1)^{n-1}S_n.$$

If there are n parallel paths with no overlapping elements between the paths, the formula has the maximum possible number of terms, 2^n-1. However, if there are overlapping elements, as is usually the case, there is extensive cancellation, so that the actual number of terms turns out to be a small fraction of the maximum number. Because the buildup alternates an inclusion, or an addition of probabilities, with an exclusion, or a subtraction, the number of terms is odd and there is always exactly one more term with a + sign than with a - sign. If the equation has 2m+1 terms, an odd number, then m terms have - signs and m+1 terms have + signs.

3.0 Sum of disjoint products

With IE the number of recursive steps is equal to the number of paths and the system polynomial is derived by alternately adding the probabilities of events with + signs and subtracting probabilities with - signs. Like IE, the sum of disjoint products (SDP) method derives a polynomial recursively, in the same number of steps as there are paths. Unlike IE, however, the formula is entirely additive: there are no exclusions and every term has a + sign.

3.1 Disjoint terms: the addition law

The addition law of probabilities is the underlying justification for SDP. If two or more events have no elements in common, the inclusive-or probability that at least one of them will occur is the sum of the probabilities of the separate sets. For example, with two events A and B, let $\overline{A}$ denote the complement not-A; and let $\overline{A}$-and-B denotes B-and-not-A. Then we have

$$P(A \text{ or } B) = P(A) + P(\overline{A} \text{ and } B).$$

Similarly with three events A, B, C

$$P(A \text{ or } B \text{ or } C) = P(A) + P(\overline{A} \text{ and } B) + P(\overline{A} \text{ and } \overline{B} \text{ and } C).$$

With n events $A_1,\ldots,A_n$ by mathematical induction

$$P(A_1 \text{ or} \ldots \text{or } A_n) = P(A_1) + P(\overline{A}_1 \text{ and } A_2) + P(\overline{A}_1 \text{ and } \overline{A}_2 \text{ and } A_3)$$
$$+ \ldots + P(\overline{A}_1 \text{ and } \overline{A}_{n-1} \text{ and } A_n). \quad (2)$$

3.2 The outer loop

There are two types of recursive steps in the Abraham version of SDP: the major step or outer loop, which is derived from the incumbent path; and a series of inner loops generated by inverting and reinverting components that were included in prior steps of the recursion, but not in the incumbent path. Each inner step results in a term that is disjoint with all other terms of the polynomial. The sum of the term probabilities for the incumbent path is the net increment of probability accounted for by the path.

Let us characterize all of the components of A, as being "one-valued" and their complementary or inverted values as "zeroes." For convenience, call the components of path A_j "Group 1" and all those of the prior paths $A_1,\ldots,A_{j-1}$ but not in A_j "Group 2." Group-1 elements are always one-valued in all of the terms generated at major step j. Group-2 elements are inverted to the zero value, and then if necessary reinverted back again to the one value. The Abraham algorithm does not require that all Group-2 components be at the zero value, nor that they all be in every term of the system reliability function.

3.3 The inner loop

At outer step j, assume there are n variables in Group 2; arrange these n variables in subscript or lexicographical order such as $x_1,\ldots,x_n$. Select a subset $x_1,\ldots,x_m$, $m \leq n$, of the n variables in Group 2 such that if every variable in the subset were inverted to its zero value, $\overline{x}_1,\ldots,\overline{x}_m$, there would be at least one zero in every one of the prior paths $A_1,\ldots,A_{j-1}$. Concatenate $\{\overline{x}_1,\ldots,\overline{x}_m\}$ with the one-valued elements of A_j; this forms the first inner term at outer step j. Each subsequent inner step both reinverts a variable from $\{\overline{x}_1,\ldots,\overline{x}_m\}$ to its one value and at the same time appends a zero-valued variable from the $\{x_{m+1},\ldots,x_n\}$ subset of Group 2.

3.4 Example

For example, with three paths $\{A_1=123,\ A_2=456,\ A_3=1789\}$, the single term formed at outer step 1 is 123. The results of the recursion are given stepwise, with each line representing an outer step and each term an inner step.

$$R_1 = 123$$

$$R_2 = R_1 + \overline{1}456 + 1\overline{2}456 + 12\overline{3}456$$

$$R_3 = R_2 + \overline{12}\ \overline{4}789 + 12\overline{3}\ \overline{4}\ 789 + 12\overline{345}789 + 12\overline{3456}789.$$

To compare SDP and IE for this case, with IE the same three paths form a probability polynomial with seven terms in three steps as follows:

$$R_1 = 123$$

$$R_2 = R_1 + 456 - 123456$$

$$R_3 = R_2 + 1789 - 123789 - 1456789 + 123456789.$$

In this particular case, with a low order of redundancy IE results in a smaller polynomial, with 7 terms, while SDP has eight terms. This is very rare; with larger systems SDP always yields a smaller polynomial, in some cases an order of magnitude smaller.

3.5 Proof of SDP

The proofs of the inner and outer loops of SDP are revised from [7]. Theorem 1 covers the inner loop and Theorem 2 the inner loop. The symbolism omits the usual zero-one structure function. Since the logic function and the probability function of SDP are 1:1 because of disjointness, the "+" sign is interchangeable either for Boolean "either-or" logic or for arithmetic addition, where the meaning is clear from the context. Likewise, multiplication can be either Boolean "and" or else an arithmetic operation.

3.5.1 Proof of the outer loop

Theorem 1: Let the system have n paths $A_1,\ldots,A_n$. The reliability is

$$R = A_1 + A_2 +\ldots+ A_n = A_1 + \overline{A}_1A_2 + \overline{A}_1\overline{A}_2A_3 +\ldots+ \overline{A}_1\ldots\overline{A}_{n-1}A_n.$$

Proof: Note the similarity to (2). The proof is by induction, using De Morgan's theorems.

$$R = A_1 + A_2 + \ldots + A_n$$

$$= [A_1+\overline{A}_1A_2+\ldots+\overline{A}_1\overline{A}_2\ldots\overline{A}_{n-2}A_{n-1}] + \overline{[A_1+\overline{A}_1A_2+\ldots+\overline{A}_1\overline{A}_2\ldots\overline{A}_{n-2}A_{n-1}]}A_n$$

$$= [\ldots] + \overline{A}_1(A_1+\overline{A}_2)\ldots(A_1+A_2+\ldots A_{n-2}+\overline{A}_{n-1})A_n$$

$$= [\ldots] + \overline{A}_1\overline{A}_2\ldots\overline{A}_{n-1}A_n. \qquad \text{QED}$$

In this proof all of the terms in the multiplication cancel except one, because of contradictions of the form $A_i\overline{A}_i$.

3.5.2 Proof of the inner loop

At each step of the inner loop, the Group-2 components are alternately inverted and then reinverted. Theorem 2 shows how to sequence the inversions. The statement of Theorem 2 is similar to Theorem 1 and the proof is a mirror image of the proof of Theorem 1

Theorem 2: Let $A = x_1,\ldots,x_i$ be a set of i 1-valued zero-one variables; then

$$\overline{A} = \overline{x}_1 + x_1\overline{x}_2 + x_1x_2\overline{x}_3 +\ldots+ x_1x_2\ldots x_{i-1}\overline{x}_i.$$

Proof, by induction, with De Morgan's theorems:

$$\overline{A} = \overline{x}_1 + \overline{x}_2 +\ldots+ \overline{x}_i$$

$$= [\overline{x}_1+x_1\overline{x}_2+\ldots+x_1x_2\ldots x_{i-2}\overline{x}_{i-1}] + \overline{[\overline{x}_1+x_1\overline{x}_2+\ldots+x_1\ldots x_{i-2}\overline{x}_{i-1}]}\overline{x}_i$$

$$= [\ldots] + x_1(\overline{x}_1+x_2)\ldots(\overline{x}_1+\overline{x}_2+\ldots+\overline{x}_{i-2}+x_{i-1})\overline{x}_i$$

$$= [\ldots] + x_1x_2\ldots x_{i-1}\overline{x}_i. \qquad \text{QED}$$

3.6 Comparing algorithms: SDP vs IE

3.6.1 Exponential property of SDP and IE

Both SDP and IE have the same property, known as "exponential time," sometimes also "NP-hard" or "NP-completeness." This terminology covers the results of "worst-case" analysis of the number of computer operations required to process very large problems. The reason these processes are exponential is that under the most extreme conditions, both the size of the resultant

formula and the computer time are exponential functions of the size of the system.

The worst case for either IE or SDP has two conditions:

a. the formula has the maximum number of terms; if there are m paths, there could be as many as 2^m-1 terms

b. each term is the maximum possible size; if there are n components in the system, each term would have n variables.

These conditions would not occur in a realistic problem. Condition b is unlikely, because the algorithms develop the formulas in such a way that some terms are smaller than others. Likewise, if every term has the same components and all of them have the same value, there would be extensive cancellation and condition a would not hold. If condition a were true, the system would be strictly parallel with no overlapping elements between paths; this could be solved more easily by series and parallel reductions rather than by overloading the computer with work that it could not process at reasonable cost.

3.6.2 Experimental results: smaller formulas with SDP than with IE

Experimental software has been developed at Oklahoma State University to run problems using both SDP and IE. The results of running five problems, reported by Chao [2], show that even though SDP has the exponential property, it is more efficient than IE in that fewer terms are generated for the system reliability formula.

case	paths	components	terms	
			SDP	IE
1	4	7	5	11
2	6	8	8	27
3	13	12	23	123
4	24	12	71	495
5	35	21	787	5287

3.6.3 Favorable cases for SDP analysis

A useful feature of SDP for certain special cases is that if the paths are ordered so that every path differs from its immediate predecessor by exactly one variable, the number of terms in the SDP reliability formula is equal to the number of paths, and the size of the formula is at most a polynomial function of the number of variables. This situation can be characterized as follows: given two successive paths A and B, with B following A, one variable in B is not in A and one variable in A is not included in B. In this case, only one variable is inverted and both the outer and inner loops consist of a single term. Two examples are described below: a strictly parallel system with n components, each component consisting of a single path, and an m-out-of-n system.

For the first example, assume there are n parallel components, $x_1, x_2, \ldots, x_n$ in a system that requires only one of these components for successful operation. Following the outer loop, Theorem 1, the SDP formula is

$$R = x_1 + \bar{x}_1 x_2 + \bar{x}_1 \bar{x}_2 x_3 + \ldots + \bar{x}_1 \bar{x}_2 \ldots \bar{x}_{n-1} x_n.$$

In this formula, there are n terms, equal to the number of paths. The number of data processing operations and computations performed is a function of n^2. This can be demonstrated by the well known sum of digits. Since the first term has one variable, the second term has two variables, etc., . . ., the n-th term has n variables, the total number of variables in the formula is

$$1 + 2 + \ldots + n = n(n+1)/2 = n^2/2 + n/2.$$

The $n^2/2$ term dominates, and the number of operations depends upon n^2.

For the second example of a favorable case for SDP analysis, an m-out-of-n system is a voting circuit for which m, $m<n$, components are required for success. If we order the paths so that every path differs from its

predecessor by exactly one component, the number of terms is equal to the number of paths. The details of this procedure are discussed in [8]. One example, a 2-out-of-4 system, is used to illustrate. Suppose the system has four components: A,B,C,D. There are six paths: AB, AC, BC, BD, AD, CD. Ths SDP formula is

$$R = AB + A\bar{B}C + \bar{A}BC + \bar{A}BC\bar{D} + A\bar{B}\,\bar{C}D + \bar{A}\,\bar{B}CD.$$

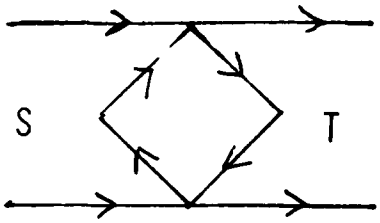
There are 6 terms, equal to the number of paths. Under worst case conditions, there could be as many as 4 variables in each term, or 24 variables altogether. The general rule is that the maximum number of variables is n^{m+1}. This is proved as follows: by the binomial counting process there are $n!/((n-m)!m!)$ terms, and each term could have n variables; hence the maximum number of variables in the system reliability formula is $n.n!/((n-m)!m!)$, or approximately n^{m+1}.

4.0 Topological reliability

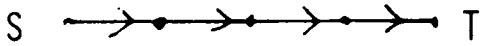
4.1 Introduction

Topological reliability (TR) was introduced in 1978 in a seminal paper by Satyanarayana and Prabhaker (S&P) to obtain a system-reliability formula by tree-oriented search. In that paper S&P demonstrated the fundamental fact of TR, that there is a 1:1 relationship between certain subgraphs, called the p-acyclic subgraphs of the system reliability graph, and the noncancelling terms of the formula. Therefore, if we can identify the p-acyclic subgraphs, we can also save processing time in deriving the formula, because there will be no duplications or cancellations.

TR decomposes the system graph by systematically stripping away edges and sequences of edges to find the p-acyclic subgraphs. Simultaneously the system reliability formula is generated. Instead of a polynomial, as in IE, the formula is in a nested and factored form that is equivalent factor by term to

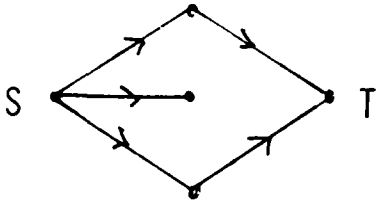


P-CYCLIC GRAPH



PATH

P-ACYCLIC GRAPH



ACYCLIC (NON P-ACYCLIC) GRAPH

FIGURE 1 EXAMPLES OF GRAPHS

the IE polynomial and follows the order of the search. Each factor also has the same sign as the corresponding IE term. The description of TR in this section follows the terminology, concepts and rules that S&P incorporated into TR, with certain minor exceptions that are noted whenever it is necessary to do so.

4.2 Concepts

4.2.1 p-graphs, p-acyclic graphs, p-cyclic graphs

A p-graph is a subgraph of the system graph G_0, with every edge of the subgraph on a path, denoting success, from the source vertex s to the terminal vertex t. An acyclic graph has no cycles; a cyclic graph has at least one cycle. A p-acyclic graph is a p-graph with no cycles; a p-cyclic graph is a p-graph with at least one cycle. Figure 1 has some examples of the different types of graphs.

4.2.2 Search tree

Tree search identifies the subgraphs of G_0, and determines how they contribute to the reliability formula. Every node of the tree represents a subgraph and every internode denotes the removal of either an edge or a sequence of edges to form a subgraph at the next node. The term "internode" is borrowed from biology; it represents the connecting line between two connecting nodes in the same branch at different levels. If the subgraph from which the string is removed is p-acyclic, the string is a "neutral" sequence and the resulting subgraph is also p-acyclic.

A search tree may be rooted at the left of the page or the right, the top or the bottom; the form S&P use is the starting point, called the root, representing G_0 at the top, and the tips of the branches, called the leaves, at the bottom. A leaf may be a path, the smallest possible type of p-graph; it can also be the end of a branch that is bypassed because of a backtracking

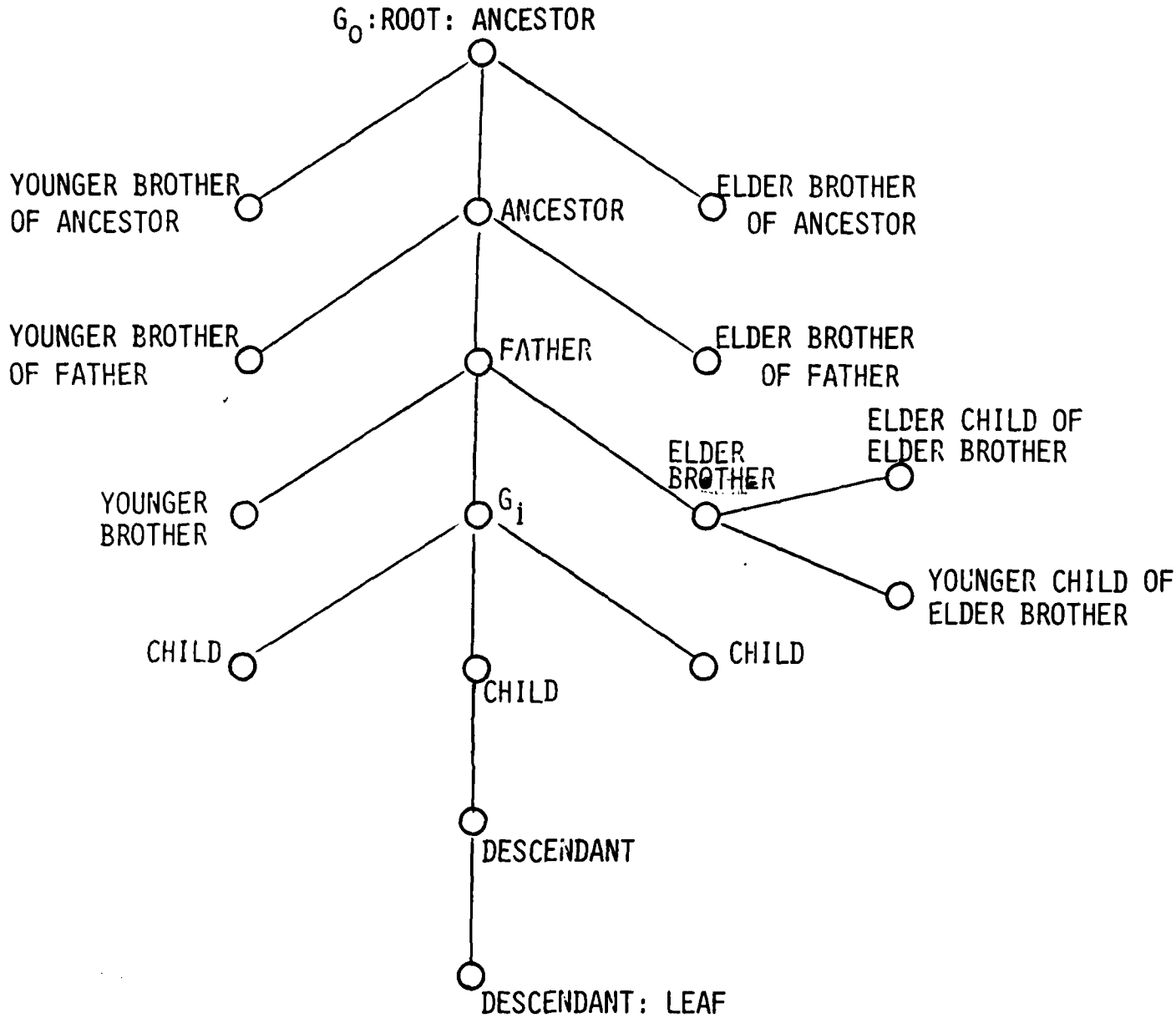


FIGURE 2, THE FAMILY TREE OF G_i

rule. The tree may be visualized as an upside-down Christmas tree with the functions inverted, the root at the top instead of the bottom, growing downward instead of upward, and with the tips of the branches at the bottom.

4.2.3 Family relationships of nodes

The descriptive family names of nodes of the search tree have a natural genealogy that follows ordinary usage of the same terms. At any given node G_1, a subgraph of G_0, the nodes obtained by branching outward and downward are children, and G_i is the father. Nodes above G_i in the same branch are ancestors and nodes below G_i in the same branch are descendants. Two or more nodes with the same father are brothers. The order of priority in processing determines which is the elder brother and which is the younger brother. In this report, the order of processing is from right to left; thus the elder brother is on the right branch and the younger brother on the left branch. Figure 2 gives the family tree of G_i.

4.2.4 Neutral sequences

A neutral sequence is a consecutive string of vertices and edges in a p-acyclic subgraph with no internal vertices connecting to other parts of the subgraph. Removing a single edge breaks the connection between s and t. In order that every child of a p-acyclic subgraph shall also be p-acyclic, only neutral sequences are removed.

4.2.5 Formations and dominations: the sign of a factor in the formula

Two related concepts from graph theory, formation and domination, help explain how the signs of the factors in the system reliability formula are obtained. Understanding these concepts provides insights into the connection between the construction of a subgraph and its contribution to the formula, as well as the procedural short cuts of TR.

A formation F of a p-graph G is a union of paths that includes all of the vertices and edges of G. F is odd, with a sign of +1, if the number of paths

in G is odd, and even, with a sign of -1, if the number of paths is even. The domination D of G is the sum of the signs of all the formations of G.

S&P showed that for a p-acyclic subgraph of G_0, D is always either +1 (i.e., there is one odd formation more than the number of even formations) or -1 (one more even formation than the number of odd formations). For a p-cyclic subgraph, D=0; a p-cyclic subgraph has the same number of even formations as odd formations. An important proof that D=0 for a p-cyclic graph was given by Willie [12].

There is a relationship between the construction of a subgraph G_i and D that makes it possible to find the sign of the factor corresponding to G_i in the system reliability formula, without identifying or counting the formations. If D=+1, the factor has a + sign; if D=-1, the factor has a - sign; and D=0 means that the subgraph is p-cyclic. The simplification is that it is only necessary to count the vertices and edges of a p-acyclic subgraph in order to find the value of D. If the difference between the number of vertices and the number of edges is odd, D=+1 and the sign of the factor is positive; if the difference is even, D=-1 and the sign is negative.

In order to find the sign, it is usually not necessary to count vertices and edges. Since the child of a p-acyclic father strips away a neutral sequence, the number of edges removed is one greater than the number of vertices removed. For example, if the neutral sequence is a single edge, no vertices are removed; if the neutral sequence is a string of two edges, the only vertex removed connects these two edges, etc. As a result, the signs of the nodes at successive levels of the search tree alternate. Once a p-acyclic ancestor is attained, the signs of all of the descendants are obtained simply just by alternating plus and minus.

4.3 Formulating the system graph: two-way edges

The system reliability graph G_0 has standard success logic; a serial connection of components in a subpath means that all the components in the subpath are needed for success, and a set of parallel connections means redundancy. A minor modification of the graph is needed when one or more of the edges is two-way. Since a two-way edge is a cycle, two one-way edges in opposite directions are substituted for each two-way edge, to facilitate subsequent decycling of the graph.

4.4 The rules of tree search

4.4.1 The weight restriction

The subgraphs and the corresponding factors of the topological reliability formula are obtained in the search, without any duplications. The principal reason there are no duplications is a rule which we shall call the weight restriction (WR). The set of edges stripped from G_0 by G_i and by the father and all of the ancestors of G_i is the weight of G_i. The WR requires that G_i may not remove any edge in the weight of the father, an ancestor or an elder brother, or of an elder brother of the father or of an ancestor. Explanation: removing an edge in the weight of the father or of an ancestor would clearly be a duplication; removing an edge in the weight of an elder brother or the elder brother of the father or an ancestor would result in duplication of a subgraph that had previously been obtained in the search.

4.4.2 The four processing rules

The processing rules of TR are numbered simply: Rule 1, Rule 2, Rule 3 and Rule 4. The rules are applied in order of priority: Rule 1 first, Rule 2 second, etc. Each rule also incorporates the WR.

Rule 1: cyclic subgraph: if G_i is cyclic, decycle G_i by removing the edges on the cycle, except for those edges in the weight of an elder brother

or of an elder brother of the father or of any ancestor. Explanation: create children by stripping edges on the cycle, one edge per child.

The corollary to Rule 1 is that if a child of a cyclic graph is cyclic, continue applying Rule 1.

Rule 2: acyclic subgraph that is not also a p-graph: at most one child can be obtained by removing edges, subject to the WR. The edges removed need not be consecutive. Explanation: the subgraph has a loose edge or edges not on a path from s to t. If all of the loose elements are removed, the child is p-acyclic.

One or more of the loose edges may be in the weight of an elder brother or the elder brother of an ancestor. Because of the WR, processing of this branch of the tree stops, and there is a backtrack to the next available node in order of priority.

Rule 3: p-acyclic subgraph with a non-p-acyclic father: if G_i is p-acyclic and the father is not p-acyclic, remove all neutral sequences except those containing edges in the weight of an elder brother or an elder brother of the father or of an ancestor. Each neutral sequence removed results in a child of G_i. Explanation: Rule 3 insures that the descendants will all be p-acyclic.

Rule 4: efficient decomposition of a p-acyclic subgraph with p-acyclic father: if G_i and the father are both p-acyclic, the internodal weights of the children (i.e., the sets of edges removed from G_i by the children) are 1:1 identical with those of the younger brothers of G_i. Explanation: all of the neutral sequences of G_i were reviously identified when the children of the father were generated. The WR is applied automatically, since G_i does not strip edges that are in the weight of an elder brother or of an elder brother of the father or of an ancestor. Since Rule 4 eliminates the need to search

for children of most p-acyclic subgraphs, there is a substantial saving of computer time.

4.4.3 Backtracking and stopping

The search has both forward and backward steps. In the forward steps, a child and the descendants of the child of an elder brother are visited before a younger brother. A backtrack takes place when a leaf is attained. A leaf may be either: 1. a path; or 2. an acyclic subgraph such that if any more edges are removed, either an open graph would result, or a violation of the WR.

Backward search follows standard backtracking procedures. The next node visited is the eldest younger brother, followed by forward search. If there are no younger brothers, move the pointer one level up the tree and to the left to the younger brother of the father, and continue forward; if the father has no younger brothers, backtrack to the younger brother of the father of the father, followed by forward search; etc. Backtracking stops when there are no ancestors with younger brothers.

4.5 Deriving the system reliability formula

4.5.1 Node labels and processing order

Every subgraph descended from G_0 has an identifying label or sequence number assigned in the forward search. The order that nodes are processed, however, is different from the sequence number, because of the search rules. All subgraphs are labelled, but a new factor for the formula is obtained only when the pointer is at a node representing a p-acyclic subgraph.

4.5.2 The system reliability formula

Let Y be the probability of success of all the elements in the system graph G_0, the product of the probabilities of success of all the components. Let G_i be a p-acyclic subgraph, d_i the domination of G_i, x_i the internodal

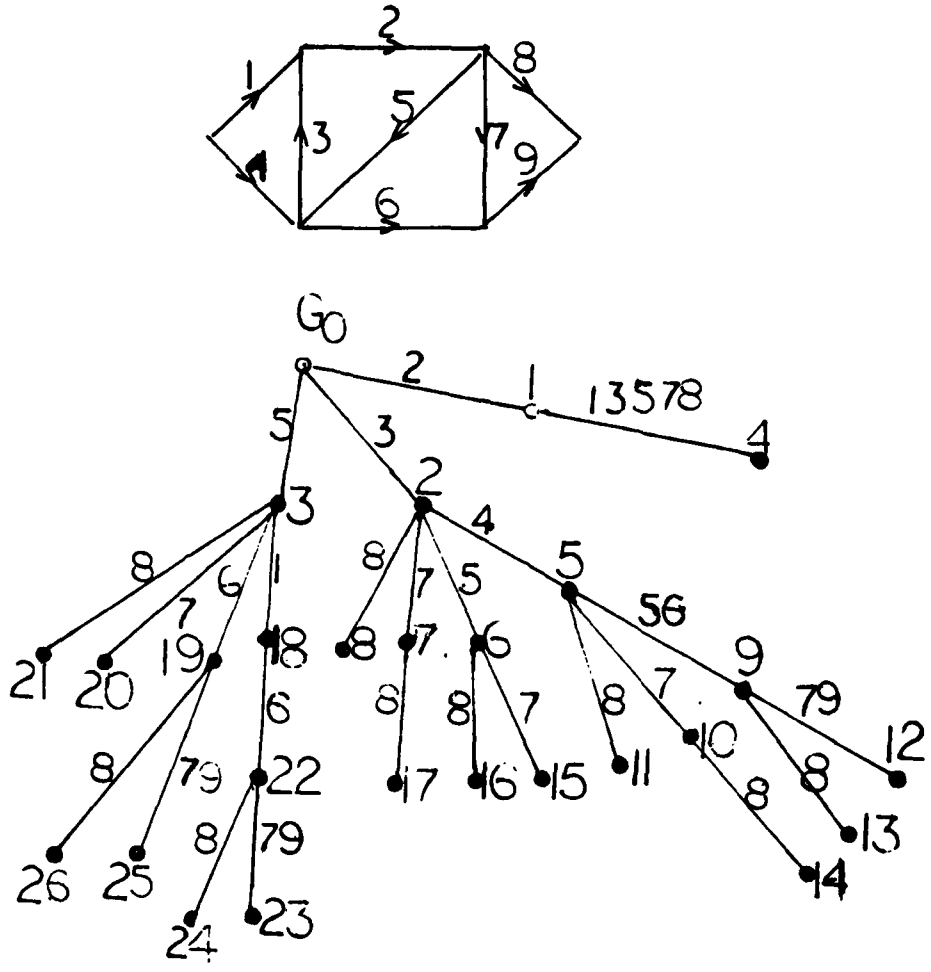
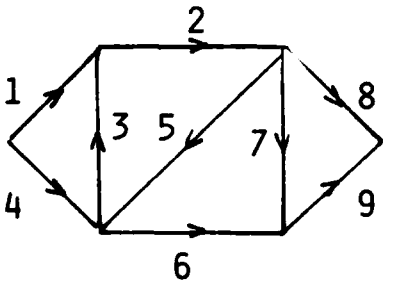


FIGURE 3, THE PARENT GRAPH G_0 AND THE SEARCH TREE



NODE LABEL	EDGES REMOVED
1	2
4	13578
2	3
5	4
9	56
12	79
13	8
10	7
14	8
11	8
6	5
15	7
16	8
7	7
17	8
8	8
3	5
18	1
22	6
23	79
24	8
19	6
25	79
26	8
20	7
21	8

FIGURE 4, DECOMPOSITION OF G_0

weight (i.e., the set of edges removed by G_i and by its non-p-acyclic father and ancestors) and $\underline{x}_i$ the product of the reliabilities of all the elements in x_i. The system formula consists of a nested set of products and sums of products of factors of the form $d+\underline{x}_i^{-1}$. The formula follows the search tree. The following example and explanation illustrate the principles involved:

$$R = Y\ ((d_1+\underline{x}_1^{-1})(d_2+\underline{x}_2^{-1}) + (d_3+\underline{x}_3^{-1})(d_4+\underline{x}_4^{-1})).$$

In this expression each of the four factors corresponds to a p-acyclic subgraph represented on the tree by a subgraph number node label. Multiplying factors

$$(d_i+\underline{x}_i^{-1})(d_j+\underline{x}_j^{-1})$$

follows from a parent-child relationship; x_j is a p-acyclic child of x_i. Adding factor products results from backtracking; the senior or eldest branch is the factor product to the left of the + sign and the junior branch is the factor product to the right.

4.6 Example

4.6.1 The search tree: node labels and subgraphs

The parent graph G_0 has 9 edges, 6 vertices, a cycle and six paths. G_0 is displayed in both Figure 3 and 4; Figure 3 has both G_0 and the search tree; Figure 4 shows the subgraphs in the order they are visited.

G_0 is cyclic; following Rule 1, we strip the edges on the cycle and form the children: G_1, G_2 and G_3, respectively denoting removal of edges 2, 3 and 5. G_1 is acyclic, not p-acyclic; Rule 2 applies: the loose edges are all removed to form G_4, the path 4-6-9. Except for G_0 and G_1, all of the other 25 subgraphs are p-acyclic.

The backtrack is to G_2, removing edge 3. G_2 is p-acyclic and the father is not p-acyclic. Rule 3 applies: all of the children and descendants are p-acyclic. Now there are significant economies in the continuation of the

search, because of Rule 4. None of the descendants of G_2 nor the descendants of any of the younger brothers remove edge 1, because of the WR. The children of G_2 are G_5, G_6, G_7 and G_8, removing 4, 5, 7 and 8. G_5, being the eldest, and the descendants of G_5, are visited before G_6, G_7, G_8 and their children.

By Rule 4, the children of G_5 have the same internodes as the younger brothers, G_6, G_7 and G_8, edges 5, 7 and 8. Removing edge 5 from G_5 would leave edge 6 loose; therefore 5-6 is a neutral sequence removed from G_5, resulting in G_9. The other children are G_{10} and G_{11}, removing edges 7 and 8.

The remainder of the branch descended from G_2 is obtained easily because of Rule 4. The branch descended from G_3 is simplified for the same reason. The simplifying principle is that when G_i and the father are both p-acyclic, the edges removed to form the children are identical to the edges removed to form the younger brothers of G_i.

4.6.2 The domination of a subgraph and the sign of a factor

The domination of a p-acyclic subgraph is either +1 or -1 and has the same numerical value as the sign of the corresponding term in the system reliability formula. There are two alternative ways to obtain the domination:

1. by counting vertices and edges: if the difference between the number of vertices and the number of edges is odd, the sign and the domination are both plus; otherwise, the difference is even, and the sign and the domination are both minus.

2. in a branch consisting only of p-acyclic subgraphs, the signs alternate at successive levels of the tree.

The first p-acyclic subgraph obtained in the search is G_4, a path with four vertices and three edges; the domination is +1. The next node visited is G_2, with 6 vertices and 8 edges; the domination is -1. The descendants of G_2 all alternate in sign at successive levels: nodes 5, 9, 12 and 13

$$R = 123456789\left(\frac{1}{123578} + \frac{1}{3}\left(-1 + \frac{1}{4}\left(1 + \frac{1}{56}\left(-1 + \frac{1}{79} + \frac{1}{8}\right)\right.\right.\right.$$

$$\left.\left.\left. + \frac{1}{7}\left(-1 + \frac{1}{8}\right) - \frac{1}{8}\right) + \frac{1}{5}\left(1 - \frac{1}{7} - \frac{1}{8}\right) + \frac{1}{7}\left(1 - \frac{1}{8}\right) + \frac{1}{8}\right)\right.$$

$$\left. + \frac{1}{5}\left(-1 + \frac{1}{1}\left(1 + \frac{1}{6}\left(-1 + \frac{1}{79} + \frac{1}{8}\right)\right) + \frac{1}{6}\left(1 - \frac{1}{79} - \frac{1}{8}\right) + \frac{1}{7} + \frac{1}{8}\right)\right)$$

FIGURE 5, THE SYSTEM RELIABILITY FORMULA FOR G_0

respectively have +, -, + and +; 10 and 14 have - and +; 11 has -; 6, 15 and 16 have +, - and -, etc. Likewise G_3 has - and all of its descendants alternate in sign.

4.6.3 The TR system reliability formula

It is a remarkable fact that the TR formula is obtained directly from the search tree. The formula is displayed in Figure 5, and it is almost self-explanatory. The initial term 123456789 is the product of the reliabilities of all the nine edges of G_0. The factor $(1234578)^{-1}$ represents G_4; the product of the G_0 initial term and this factor is the product of the reliabilities of the three edges in the path 4-6-9. The + sign preceding 3^{-1} corresponds to the backtrack to G_2, removing edge 3. G_2 has a sign of -1; this is the first term of the next factor. The eldest child of G_2 is G_5, removing edge 4; this results in the factor 4^{-1}; since the signs alternate, the domination is +1.

The process of generating factors and their signs continues: the factors are the reciprocals of the products of the probabilities of the edges removed; multiplications are parent-child relationships; additions are backtracks; and the signs alternate.

Another notable aspect of the TR formula is the fact that fully expanded, it is identical to the IE formula for G_0. There are 25 terms, each term representing a p-acyclic subgraph, and the signs of the terms in IE are identical to the signs of the factors in TR.

5.0 Extensions of topological reliability: the unified approach

5.1 Introduction

We have shown in the foregoing sections of this paper that TR derives the reliability formula of a coherent system that has a single source and a single terminal.

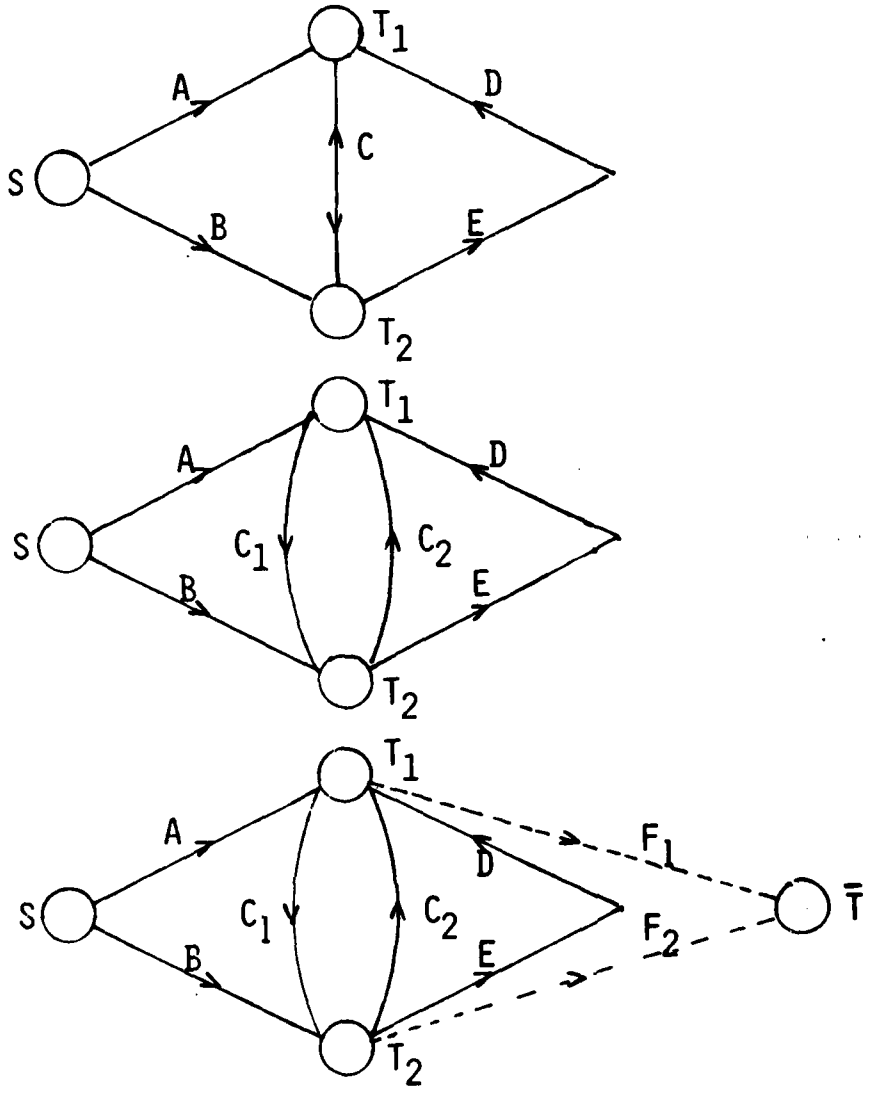


FIGURE 6, S3T,

REPLACEMENT AND SUPERGRAPH G_0

In 1982 Satyanarayana [9] showed that by modifying the graph, it is possible to efficiently develop reliability formulas for systems that are more complex than just s-t, by TR. Examples are: systems with every vertex being both a source and a terminal in a two-way connection to every other vertex--this is called "overall reliability;" and systems with mixtures of one- and two-way connections. One special case that occurs in communications networks is the "k-terminal problem": a subset of k of the vertices are compulsory recipients of every message, with two-way communications between every member of the k-subset, the other vertices primarily serving the function of relays.

The key idea that makes this type of structural solution of a generalized reliability problem is the "unified approach"--creating a supergraph with both a dummy source $\bar{s}$ and a dummy terminal $\bar{t}$, if necessary, and dummy edges connecting members of the k-subset and $\bar{s}$ and $\bar{t}$ in such a way that a system reliability formula is generated by TR in the same way it is obtained for coherent s-t systems. The dummy edges are elder brothers to the rest of the graph, and hence are never removed to form subgraphs, because of the weight restriction.

5.2 Example: source-to-k-terminal (SKT) problem

The example is a four-vertex five-edge diamond shaped graph that differs from a bridge circuit in that it has two terminals rather than one, both terminals in the middle instead of at the right-hand edge, and one edge is directed from right to left instead of from left to right. We shall call this a K-graph, S3T, with K = 3. Since edge c goes both ways, it is replaced by two one-way edges, c_1 and c_2, in opposite directions. Both the original graph and the replacement are shown in Figure 6.

The system reliability problem is to determine the probability that a message received at the source vertex s at the left-hand edge will be

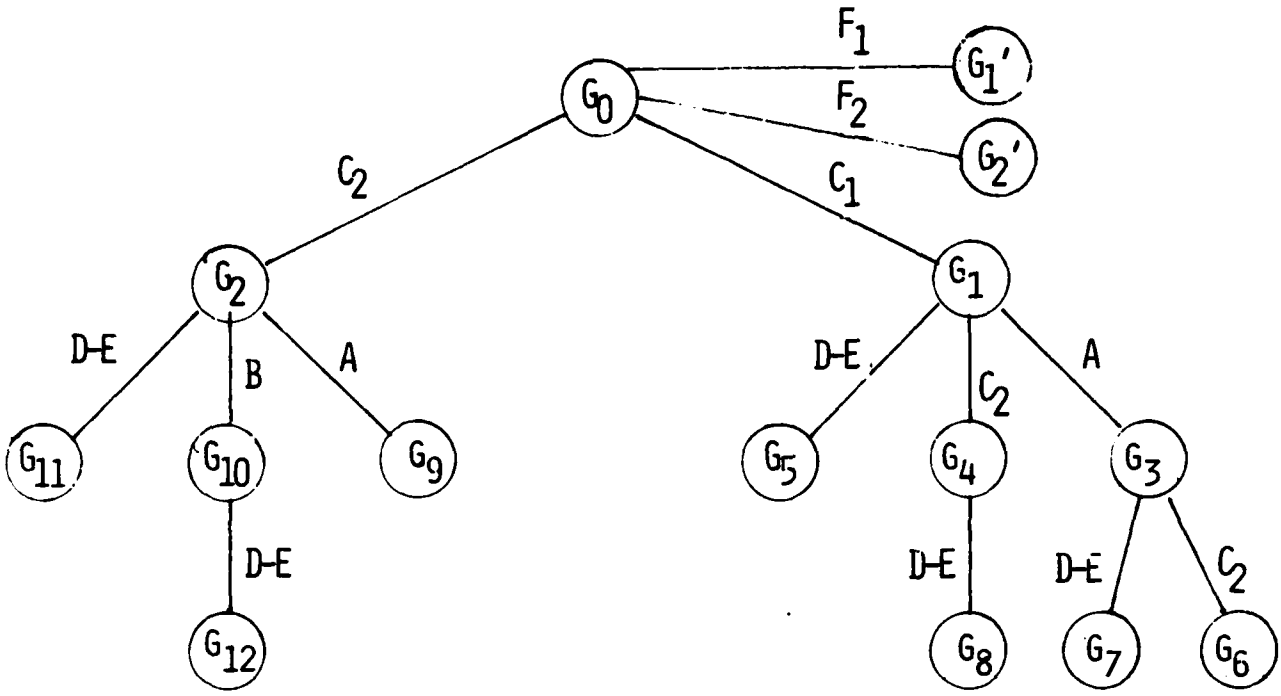


FIGURE 7, S3T SEARCH TREE

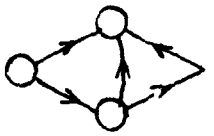
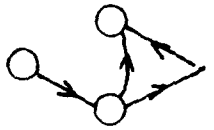
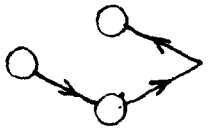
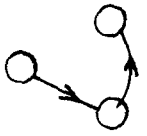
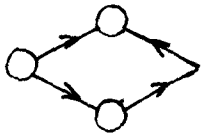
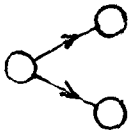
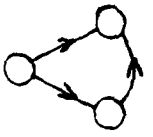
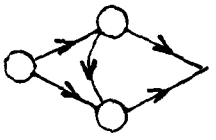
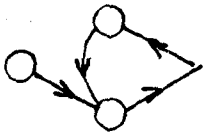
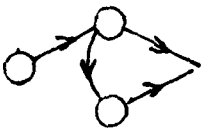
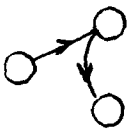
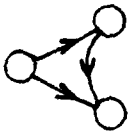
		WEIGHT	EDGES	VERTICES	D_i
G_1		C_1	5	4	+1
G_3		A	4	4	-1
G_6		C_2	3	4	+1
G_7		D-E	2	3	+1
G_4		C_2	4	4	-1
G_8		D-E	2	3	+1
G_5		D-E	3	3	-1
G_2	 K-CYCLIC	C_2			0
G_9	 K-CYCLIC	A			0
G_{10}	 K-CYCLIC	B			+1
G_{12}		D-E	2	3	+1
G_{11}		D-E	3	3	-1

FIGURE 8, DECOMPOSITION OF S3T
DUMMY EDGES F_1 AND F_2 AND DUMMY TERMINAL $\bar{T}$ DELETED.

communicated to both terminals, t_1 and t_2. In order to solve this problem by topological methods, we form a supergraph, shown in the bottom third of Figure 6, together with an artificial terminal $\bar{t}$ and the dummy edges f_1 and f_2 that connect t_1 and t_2 to $\bar{t}$. Call the supergraph G_0, and decompose it in a search for the "K-acyclic" subgraphs such that every edge is on a path from s to $\bar{t}$, including also both terminals t_1 and t_2. The search tree, including the dummy edges, is shown in Figure 7 and the subgraphs corresponding to the nodes of the tree, with the dummy edges deleted, are displayed in Figure 8.

The first two children of G_0, G_1, and G_2,, result from removing the dummy edges; thus they are elder brothers to every ancestor of every node of the tree and the dummy edges are not removed again because of the weight restriction. This guarantees that both terminals t_1 and t_2 are included in every K-acyclic subgraph.

G_0 is cyclic; following Rule 1 it is decycled by removing edge c_1 to form G_1, and c_2 is removed to form G_2. G_1 is K-acyclic; following Rule 3, remove the neutral sequences a, c_2 and d-e, forming the children G_3, G_4 and G_5, which are also K-acyclic. Following Rule 4, the children of G_3, G_6 and G_7, have the same internodal weights as the younger brothers of G_3, c_2 and d-e. Likewise the child of G_4, G_8, has the same internodal weight as G_5, d-e.

The other principal branch of the tree, starting with the removal of c_2, yields a mixture of K-cyclic and K-acyclic subgraphs. G_2 is K-cyclic; by the corollary to Rule 1, the decycling continues by stripping the edges a, b and d-e, forming the children G_9, G_{10}, and G_{11}. G_9 is K-cyclic, but it cannot be decomposed any further without breaking the connection from s to $\bar{t}$; therefore it is necessary to backtrack. G_{10} is K-cyclic; however, its child, G_{12}, by removing d-e, is K-acyclic. Likewise, the last child of G_2, G_{11}, obtained by removing d-e, is K-acyclic. The formula for S3T is given in Figure 9.

$$R = ABC_1C_2DE\left(\frac{1}{C_1}\left(1+\frac{1}{A}\left(-1+\frac{1}{C_2}+\frac{1}{D\cdot E}\right)\right.\right.$$

$$\left.\left.+\frac{1}{C_2}\left(-1+\frac{1}{D\cdot E}\right)-\frac{1}{D\cdot E}\right)+\frac{1}{C_2}\left(-1+\frac{1}{BDE}-\frac{1}{DE}\right)\right)$$

FIGURE 9, THE FORMULA FOR S3T

6.0 Complexity analysis

An important aspect of evaluating computer programs and algorithms and other types of data processing systems is frequently called "complexity analysis;" we shall define it as "classifying an algorithm into a category of difficulty, based on the maximum number of computer operations required to obtain a solution." Three types of complexity are discussed in the literature, in increasing order of the number of computer operations required: linear time, polynomial time and exponential time or NP-completeness.

1. Linear time implies that the number of computer operations does not exceed a linear function of the system size n (i.e., the number of paths or variables, or the number of edges or vertices or both the edges and the vertices, etc.). Linear time is obviously desirable, but it is usually unattainable in practice, because computer time invariably grows faster than the system size.

2. Polynomial time means that under worst-case conditions, the number of operations does not exceed a polynomial function of n; problems are further classified according to the order of the polynomial function: n^2, n^3, etc. Polynomial time is desirable, and frequently it is attainable in practice, if not with worst cases, because of short cuts that are built into the algorithm, or because the analyst modifies the problem in such a way that reasonable amounts of computer time are needed. Refer to the examples in Section 3.6 of this paper that describe ways of organizing the paths in certain special cases to make SDP operate in polynomial time.

3. NP-completeness means that for the worst case, the number of operations is an exponential function of n. Since a data structure being processed is not always at its worst case, a different way of defining NP-completeness is it is not possible to guarantee that the algorithm being

processed will not exceed polynomial time. The fact that the worst case may be exponential, however, does not necessarily mean that the computer time grows without limits. First, the problem usually is not the worst case; second, the analyst may have the option of simplifying and reorganizing the data so as to keep the computer time within reason.

6.1 Comparing TR and IE

6.1.1 The worst case

In the foregoing section 3.6, we discussed the exponential-time property of inclusion-exclusion, in comparison to that of the Abraham SDP algorithm. The worst case has two conditions:

a. the number of terms in the system reliability formula is an exponential function of the number of paths

b. the size of the formula is an exponential function of the number of components in the system.

IE and TR both generate the same formula, which is a polynomial in IE and a nested and factored form in TR, the TR factors being 1:1 with the IE terms and having the same signs. Even though the resulting computed value of the system reliability would be the same for either version of the formula, there is a great difference not only in the appearance of the two different versions, but also in the way the forms are obtained. With respect to the counting of terms, whereas IE might duplicate a given term many times, and then cancel it out every time it appears after the first time, TR obtains the corresponding factor just once, with no duplications. Thus TR is linear in the number of p-acyclic subgraphs, though not in the number of paths; IE has no comparable property.

The size of the formula is smaller with TR than in IE, because the number of variable entries is a fraction of the number of entries in IE. Every time

a TR factor is generated, the formula is augmented by a product involving a single component, or at most the product of elements of a neutral sequence. By contrast, the corresponding IE term could be of size n, where n is the number of variables in the system.

6.1.2 TR uses less storage and does not process cycles

TR also uses less storage than IE. With IE, it is necessary to store every term, in order to cancel out terms that are zeroed out. Since these cancellations represent cycles in the system graph, computer time is wasted processing cycles that do not enter the formula. In this respect TR is much more efficient than IE. Cycles are never processed because of Rule 1, and the factors that enter the formula represent only the noncancelling p-acyclic subgraphs. Only the factor at the incumbent position of the pointer has to be stored, and a minimal amount of additional information, such as the value of a pre-existing factor that is the result of all prior data processing and calculations.

6.1.3 Advantages of TR

The major advantages of TR have been discussed in the foregoing: shorter formula, less storage, much less computer time and the ability to process a wider variety of large and complex networks. Another advantage is that it is unnecessary to find the paths before starting the calculations, as is the case with IE.

6.1.4 NP-completeness

Although TR is more efficient than conventional IE, it has the same NP-completeness property that both SDP and IE have. The fact that an algorithm has this property does not mean that large, complicated systems cannot be solved with reasonable computer time. Many exponential algorithms, for example, the simplex method of linear programming, are solved by computers

with great economic benefit to the user, because the speed of the computer and its accuracy as compared to that of human calculation, makes it worthwhile to use automatic calculation, even though there is theoretically virtually no limit to the time required, for a worst case that does not occur in practice. As a further guide to the theory of useful NP-complete algorithms, refer to the interesting book by Garey and Johnson [5].

7.0 References

[1] J. A. Abraham, "An improved method for network reliability," IEEE Trans. Reliability, vol R-28, 1979 Apr, pp 58-61.

[2] C. P. Chao, "Recursive disjoint products: the reliability evaluation for both coherent and noncoherent systems," MBA report, Oklahoma State University, Department of Management, 1983 Jun.

[3] W. Feller, An Introduction to Probability Theory and its Applications, vol 1, 3rd edition, John Wiley & sons, inc, 1968.

[4] L. Fratta, U. G. Montanari, "A Boolean algebra method for computing the terminal reliability in a communication network," IEEE Trans. Circuit Theory, vol CT-20, 1973 May, pp 203-211.

[5] M. R. Garey, D. S. Johnson, Computers and Intractibility: A Guide to the Theory of NP-Completeness, Bell Telephone Laboratories, Inc., 1979.

[6] M. O. Locks, "Relationship between minimal path sets and cut sets," IEEE Trans. Reliability, vol R-27, 1978 Jun, pp 107-109.

[7] M. O. Locks, "Recursive disjoint products: a review of three algorithms," IEEE Trans. Reliability, vol R-31, 1982 Apr, pp 33-35.

[8] M. O. Locks, "Comments on: improved method of inclusion-exclusion applied to k-out-of-n systems," IEEE Trans. Reliability, forthcoming.

[9] A. Satyanarayana, "A unified formula for analysis of some network reliability problems," IEEE Trans. Reliability, vol R-31, 1982 Apr, pp 23-32.

[10] A. Satyanarayana, A. Prabhaker, "New topological formula and rapid algorithm for reliability analysis of complex networks," IEEE Trans. Reliability, vol R-27, 1978 Jun, pp 82-100.

[11] United States Nuclear Regulatory Commission, Reactor Safety Study: An Assessment of Accident Risks in U.S. Commercial Nuclear Power Plants, WASH-1400, (Nureg-75/014), 1975 Oct.

[12] R. R. Willie, "A theorem concerning cyclic directed graphs with applications to network reliability," Networks, vol 10, 1980, pp 71-78.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) OSU OBER 85-1	5. MONITORING ORGANIZATION REPORT NUMBER(S) AFOSR-TR- 85-0094
6a. NAME OF PERFORMING ORGANIZATION Oklahoma State University 6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Air Force Office of Scientific Research
6c. ADDRESS (City, State and ZIP Code) Office of Business and Economic Research Stillwater OK 74078	7b. ADDRESS (City, State and ZIP Code) Directorate of Mathematical & Information Sciences, Bolling AFB DC 20332-6448
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR 8b. OFFICE SYMBOL (If applicable) NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-82-0251
8c. ADDRESS (City, State and ZIP Code) Bolling AFB DC 20332-6448	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. 61102F; PROJECT NO. 2304; TASK NO. A5; WORK UNIT NO.

11. TITLE (Include Security Classification)
SOME RECENT DEVELOPMENTS IN SYSTEM RELIABILITY: COMPARATIVE STUDY OF INCLUSION-EXCLUSION, SUM OF DISJOINT PRODUCTS AND TOPOLOGICAL RELIABILITY. FINAL SCIENTIFIC REPORT. GRANT /AFOSR-82-0251, 1 JULY 1983 - 30 SEPTEMBER 1984.

12. PERSONAL AUTHOR(S)
Mitchell O. Locks

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Yr., Mo., Day)	15. PAGE COUNT
Final	FROM 1/7/83 TO 30/9/84	JAN 85	42

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
System reliability; inclusion-exclusion; sum of disjoint products; topological reliability; m-out-of-n; source-to-multiple terminal reliability.

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

System reliability analysis calculates the probability of success for a system, based on the component reliabilities and the configuration. First, a logic function is obtained in the form of either a tree, chart, graph, diagram or list of paths. From this logic function a probability formula is derived. The classical or conventional method of generating a formula is inclusion-exclusion (IE).

With the past decade there have been some significant new developments that resulted in ways to estimate the system reliability that are more efficient than IE. Two of these techniques are discussed in this paper: sum of disjoint products (SDP), and the topological reliability (TR) of Satyanarayana and Prabhaker (S&P). This paper covers the theory and procedures of both techniques, shows their interrelationships with IE, and discusses complexity considerations and computer time needed for preparation of a system formula. The discussion on TR also includes advanced applications such as overall (CONTINUED)

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE NUMBER (Include Area Code)	22c. OFFICE SYMBOL
MAJ Brian W. Woodruff	(202) 767- 5027	NM

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

ITEM #19, ABSTRACT, CONTINUED: reliability and k-terminal reliability, classes of problems that can conveniently be solved by TR with minor modifications of the logic.

END

FILMED

4-85

DTIC